Nuclear Instruments and Methods in Physics Research A 356 (1995) 507–513

NUCLEAR INSTRUMENTS & METHODS IN PHYSICS RESEARCH Section A

ELSEVIER

A neural network approach for the TPC signal processing

P. Cennini [a], S. Cittolin [a], J.P. Revol [a], C. Rubbia [a], W.H. Tian [a], D. Dzialo Giudice [b], X. Li [b], S. Motto [b], P. Picchi [b], P. Boccaccio [c], F. Cavanna [d], G. Piano Mortari [d], M. Verdecchia [d], D. Cline [e], S. Otwinowski [e], G.H. Wang [e], M. Zhou [e], A. Bettini [f], F. Casagrande [f], S. Centro [f], C. De Vecchi [f], A. Pepato [f], F. Pietropaolo [f], S. Ventura [f], P. Benetti [g], E. Calligarich [g], A. Cesana [g], R. Dolfini [g], A. Galli Tognota [g], A. Gigli Berzolari [g], F. Mauri [g], L. Mazzone [g], C. Montanari [g], M. Negrini [g], A. Piazzoli [g], A. Rappoldi [g], G.L. Raselli [g,*], D. Scannicchio [g], M. Terrani [g], C. Vignoli [g], L. Periale [h], S. Suzuki [h]

[a] *CERN, CH-1211, Geneva 23, Switzerland*
[b] *Lab. Naz. di Frascati dell'INFN, via E. Fermi 40, Frascati (Roma), Italy*
[c] *Laboratori Nazionali di Legnaro dell'INFN, Italy*
[d] *Dipartimento di Fisica e INFN, Università dell'Aquila, via Vetoio, Coppito (AQ), Italy*
[e] *Department of Physics, UCLA, Los Angeles, CA 90024, USA*
[f] *Dipartimento di Fisica e INFN, Università di Padova, via Marzolo 8, Padova, Italy*
[g] *Dipartimento di Fisica Nucleare e Teorica e INFN, Università di Pavia, via Bassi 6, Pavia, Italy*
[h] *Istituto di Cosmo Geofisica del CNR di Torino, corso Fiume 4, Torino, Italy*

Received 13 October 1994

Abstract

Artificial neural networks may in some cases be alternatives to programmed computing. Since they offer a new important approach to information processing, we have investigated if the accuracy offered by this technique is good enough to extract physical information from the signals coming from a liquid argon time projection chamber. The results obtained implementing a neural network algorithm on a sequential scalar computer have been compared to those of a standard best-fit procedure on the same machine. This new method appears to be suited for the analysis of the events occurring in a very large detector, as that foreseen for the ICARUS experiment.

1. Introduction

We have constructed and operated at CERN a 3 ton liquid argon time projection chamber (LAr TPC) within the R&D programme for the ICARUS project [1]. Beginning in June 1991 we started to collect events from cosmic rays and from radioactive sources to measure the relevant physical parameters of the detector, to understand its response to ionizing events and to gain operating experience over a long period of time.

So far, the way we have followed to extract physical quantities from raw data is to perform a least-squares fit of the signal using a theoretical analytical function. Since this procedure needs a long time to minimize over the function parameters, the possibility to use faster computational methods has been investigated.

In this paper we present a short analysis of the essential features of an approach based on the implementation of a neural network algorithm on a sequential scalar computer. The paper starts with a brief description of the data analysis based on the best-fit procedure. We then describe the neural network approach, its software implementation and the learning process. We finally conclude presenting and discussing the results obtained using different methods of analysis and a possible use for the ICARUS experiment at the Gran Sasso laboratory.

2. The LAr TPC data analysis

The construction of the 3 ton detector and the complete readout system of the LAr TPC are described in a separate paper, where some bubble chamber grade event images are also shown [2].

* Corresponding author. E-mail: raselli@pv.infn.it.

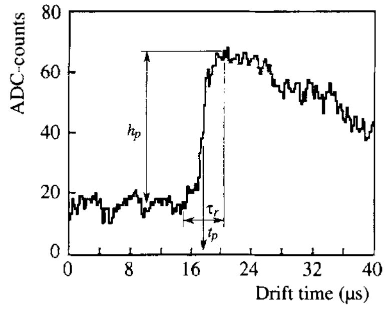

Fig. 1. A LAr TPC collection signal with the parameter symbols: pulse position in time (t_p), pulse height (h_p) rise time (τ_r).

The collection charge signal from each TPC readout channel is sampled with 8-bit flash ADCs at a frequency (5–20 MHz) that provides several measurements for each pulse and the resulting data are stored in raw data files containing 2048 samples times 192 channels per event.

These signals carry three main quantities (Fig. 1) which are relevant for the physical analysis:

t_p pulse position in time, assumed as the point of inflection of the signal leading edge, which contains information on the drift coordinate;

h_p pulse height, proportional to the collected charge;

τ_r rise time, which depends both on the track dip angle with respect to the collection plane, and on the diffusion of the electron cloud during the drift.

The parameters are extracted in two steps: first, the valid regions (those which contain some signals well above the noise) inside the huge event data are traced and an associated hit table is created (hit finding procedure); then, a least-squares fit is locally performed on each single region traced on the table, to recover the signal parameters (best-fit procedure).

The hit finding algorithm returns, for each channel, a vector containing 100 consecutive samples (region of interest) approximately centered on the point of inflection of the signal leading edge.

For the best-fit calculation we use an analytical function with 6 parameters that reproduces relatively well a wide range of signal shapes.The parameters are: pulse height (h_p), pulse position in time (t_p), rise time (τ_r), amplifier decay time (τ_d) and include a linear baseline ($a + bt$). This function is defined by:

$$f(t) = h_p \frac{\exp\left(\dfrac{t_p - t}{\tau_d}\right)}{1 + \exp\left(\dfrac{t_p - t}{\tau_r}\right)} + a + bt. \qquad (1)$$

The results based on the analysis of the data using this method, presented in Ref. [3], were used to study some important physical parameters of the detector: free electron yield, recombination probability, drift velocity, diffusion coefficient, free electron lifetime, spatial resolution and their possible dependence on the electric field intensity.

3. The neural network approach

3.1. Neural network simulation

An artificial neural network can be thought of as a sort of "black box" processing system; its operational capabilities allow the reproduction of an application $x \xrightarrow{f} y$ between two sets of vectors. These sets represent, in the present case, all the possible signal shapes and the physical parameter values.

The information stored inside the network, that allows the desired application to be performed, is not directly-meaningful, since it is developed in adaptive response to the specific problem.

According to the Hecht–Nielsen theorem [4], any analytical application can be computed with high accuracy using a feed-forward three-layer non-linear network with full connection between adjacent layers.

A neural network of this type acts in quite a simple way. The basic processing units are the neurons which exchange information each other by mean of synapses. Each ith neuron has its own specific activation A_i which is presented to the output O_i by means of a non-linear transfer function. The sigmoid function

$$O_i = S(A_i) = \frac{1}{1 + \exp(-A_i)} \qquad (2)$$

is widely used and the output value of each neuron ranges between 0 and 1. Neurons are organized in three consecutive layers called the input, the hidden and the output layer respectively. Adjacent layers are linked by mean of a numerical coefficient matrix that determines the "synaptic strength", the weight, of each connection. For the input layer the activation comes directly from the outside. In the hidden and in the output layer, the activation is the sum of the values coming from the neurons directly connected multiplied by the corresponding weight. The values transferred to the output of the last layer are the result of the network process.

The process of extracting physical information using this type of neural network architecture is straightforward: each signal is presented to the inputs of the network, which returns, as output, the relevant quantities for the physical analysis. The weights have to be carefully determined in such a way that the network could reproduce the desired application. They are adjusted using input and target data by means of an iterative learning process called training.

The highly dynamic development of neural networks in recent years have stressed the need for hardware implementations; at present VLSI general-purpose neural network chips are commercially available. However, training a network like this will involve some difficulties if the connections between neurons are made up of physical components with specific values. This problem may be solved by using any kind of sequential simulation software that handles the processing of the neurons.

For this purpose we use a workstation DEC 3000/800 AXP for network simulation. This is based on a 64-bit RISC CPU with a very high clock rate (200 MHz). We have developed a FORTRAN 77 language program which performs the simulation of the neural process [5]. The architecture of a neural network can easily be defined or modified simply determining the number of neurons in each layer. If the architecture fits the specific problem, the network can be trained by the program in such a way that it could reproduce the desired application. The calculated weights are stored in a ASCII file.

3.2. Network architecture

In the present case the network should be able to receive, at the input layer, a number of ADC samples large enough to fully contain the signal leading edge for all the possible values of the rise time that may appear in the data. This means 100 points if we assume 10 MHz as ADC sampling frequency. The network response consists of the three quantities that are relevant for the TPC waveform analysis (pulse height, position in time, rise time); this means that we need at least three neurons in the output layer. Since the amplifier decay time and the baseline slope values affect the measurement of the previous three quantities, they are used as additional input parameters of the network. The value of the baseline slope can be directly measured during the hit-finding procedure whereas the amplifier decay time depends only on the electronic readout chain and its value is fixed by direct measurement with the test pulse data [3].

In short, the architecture of our network consists of 102 neurons in the input layer and 3 in the output one. With respect to the number of neurons in the hidden layer, we started with 200 neurons. Later, to speed-up the network processing simulation, we lowered this number and in the final configuration we used 50 neurons.

3.3. Training process

There is a variety of learning paradigms in the literature (see Ref. [4,6]) depending on the network architecture as well as on the characteristic of the input/output data. In the present case we have followed the scheme known as back-propagation.

The training phase is organized as follows: by means of a Monte Carlo program we simulate a set of 1000 signals using uniform distributions of the function parameters. The range of variation, shown in Table 1, is chosen in order to reproduce the different signal shapes as in real data.

Table 1
Range of variation of the function parameters used to reproduce the different signal shapes as in real data

Parameter	Min value	Max value
Pulse height	10 ADC-counts	200 ADC-counts
Rise time	0.25 μs	2.5 μs
Decay time	30 μs	50 μs
Baseline pedestal	0 ADC-counts	100 ADC-counts
Baseline slope	−1 ADC-counts/μs	+1 ADC counts/μs
Noise	0 ADC-counts	+4 ADC-counts

The Monte Carlo program performs a detailed signal simulation and returns, for each signal, a vector containing 100 consecutive values broadly centered ($\sigma = 0.2$ μs) on the inflexion point of the signal leading edge, together with the parameter values used as input [5]. The time spread takes into account the time resolution of the hit finding procedure. Using the data contained in the vector, including the values of the amplifier decay time (τ_d) and the baseline slope (b), a pattern of activation for the network inputs is created.

These patterns are repeatedly applied to the network; a complete cycle is called an epoch. At each iteration k, an error signal ρ_j is computed for each jth output, based on the difference between the output value and the true value coming from the simulation. The energy function of the network is defined as the sum of the errors squared

$$E_k = \tfrac{1}{2} \sum_j \rho_j^2. \qquad (3)$$

Its value is propagated backwards through the network to modify each weight proportionally, in order to be minimized. This is called gradient descent procedure [6]. The updating equation is

$$\Delta w_{ij}(k) = -\varepsilon \frac{\partial E_k}{\partial w_{ij}} + \alpha \Delta w_{ij}(k-1), \qquad (4)$$

where the proportionality constant ε is called learning rate ($0 < \epsilon < 1$) and α is the momentum factor ($0 < \alpha < 1$). If random noise is added, its perturbation of the gradient descent algorithm is often enough to prevent the system from falling into a local minimum.

The training starts using relatively high values of the learning rate, momentum and noise; the factors are gradually changed during the training over several thousands of epochs. An example of the evolution of the energy function during the training, for the architecture with 50 neurons in the hidden layer, is shown in Fig. 2. A training session like this one takes about 150 hours of CPU time on the above mentioned AXP computer.

4. Results

Two different signal sets, one containing 10^4 Monte Carlo signals and one consisting of about 5×10^4 signals

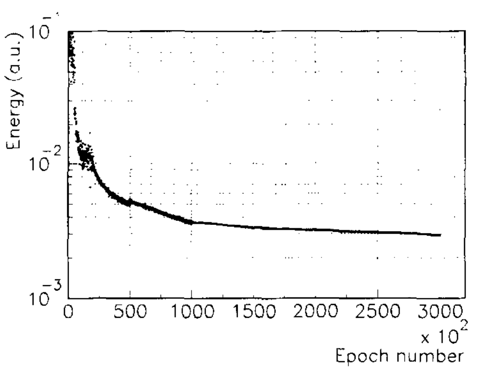

Fig. 2. Evolution of the energy function during the training of the neural network using 50 neurons in the hidden layer.

coming from real data acquisitions are used to test and compare the response of the network with that of the best-fit technique using the CERN Minuit package [7]. The first sample is used to evaluate the linearity and the intrinsic network resolution; the second one is used to compare physical results obtained with the two different approaches.

4.1. Pulse height response

In Fig 3a the value of the pulse height returned by the network (50 neurons in the hidden layer) is plotted versus the value used as input in the Monte Carlo simulation program. The response of the network is linear and has a uniform resolution over the whole range of variation of the parameter. To evaluate the intrinsic resolution, the error distribution is plotted and then fitted with a gaussian line shape (Fig. 3b). The resolution is about 0.73 ADC counts that corresponds, assuming a gain of 330 electrons per ADC-count (like in the prototype readout system), to about 240 electrons. This result is well below the value of the electronic noise that affects the normal TPC data ($\approx$ 1300 electrons RMS, see Ref. [3]).

4.2. Rise time resolution

In Fig 4a the value of the rise time returned by the network (50 neurons in the hidden layer) is plotted versus the value used as input in the Monte Carlo simulation program. The response of the network is again linear and has an uniform resolution over the whole range of variation of the parameter. Its intrinsic resolution, shown in Fig. 4b ($\sigma < 0.01$ μs) is very good if compared to the rise time spread due to the electron longitudinal diffusion (ranging from 0.5 to 0.7 μs for an electric field of 350 V/cm [3]).

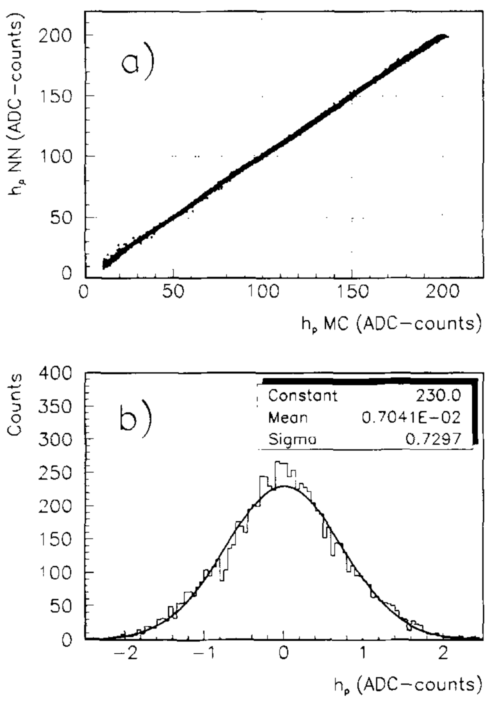

Fig. 3. Capability of the network to get the pulse height (h_p) of the input signal (NN = neural network response, MC = Monte Carlo input value).

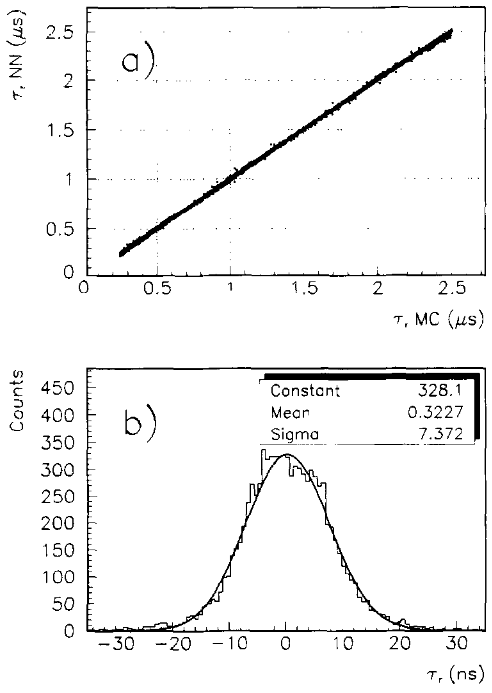

Fig. 4. Capability of the network to get the rise time (τ_r) of the input signal (NN = neural network response, MC = Monte Carlo input value).

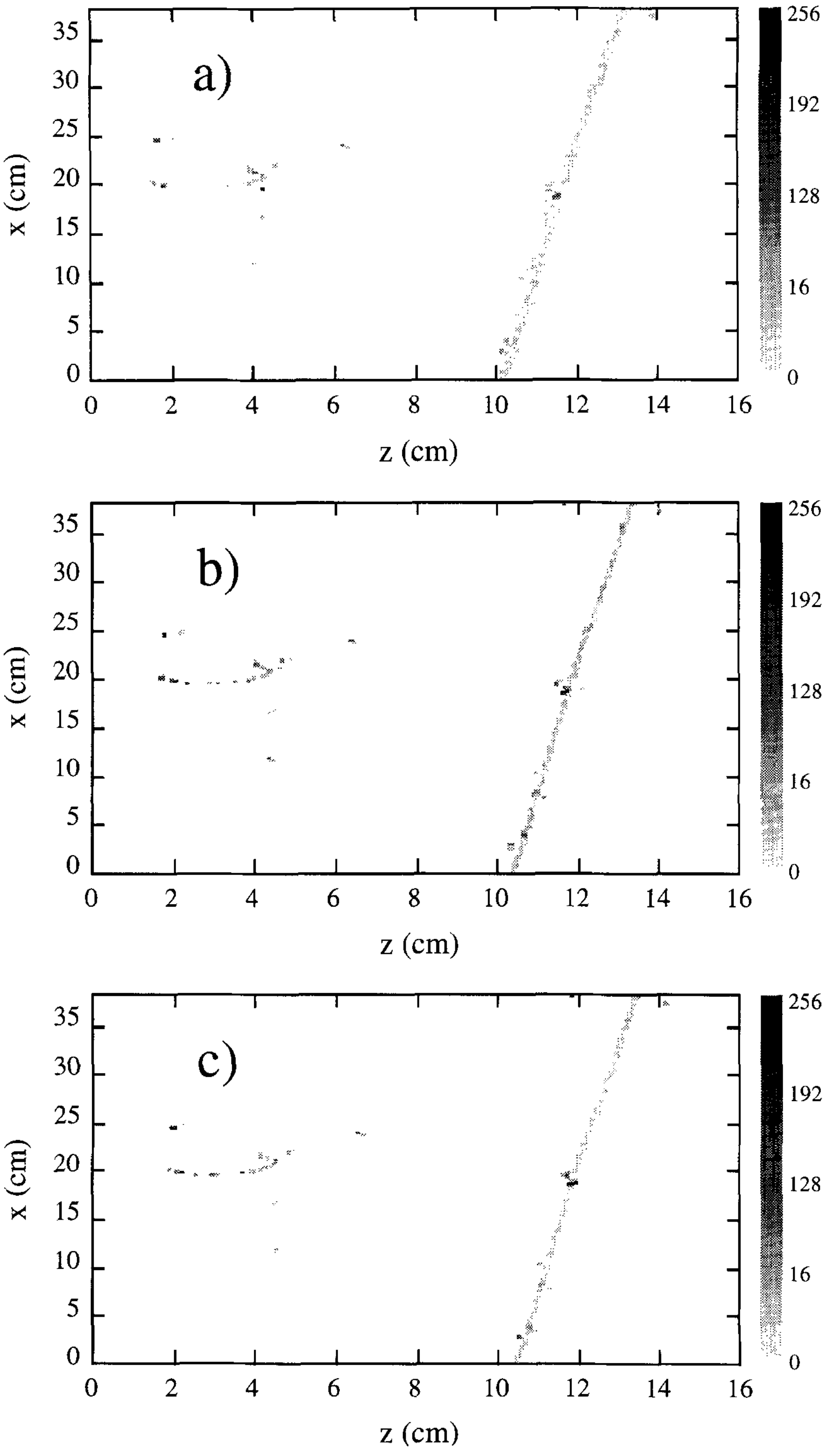

Fig. 5. The image of a cosmic-ray track (muon) processed using tree different algorithms: (a) the hit-finding filter; (b) neural network; (c) best-fit. The two coordinates x and z are proportional to the TPC channel number and to the drift time respectively. The grey level of the pixel codes the pulse height (ADC-counts).

Table 2
Single point spatial resolution obtained with different approaches (h.u. is the number of neurons in the hidden layer)

Hit finding procedure	331 ± 3 μm
Hit finding procedure + neural network (200 h.u.)	192 ± 3 μm
Hit finding procedure + neural network (100 h.u.)	195 ± 3 μm
Hit finding procedure + neural network (50 h.u.)	208 ± 3 μm
Hit finding procedure + best fit	142 ± 3 μm

4.3. Spatial resolution

The evaluation of the capability of the network to get the pulse position of the input signal can be made measuring the TPC single point spatial resolution along the drift direction. Using vertical muon events, the arrival time resolution is given by the width of the distribution of the residuals to a straight line fit made on three contiguous channels at the time, to minimize the contribution from multiple scattering [3]. From the knowledge of the electron drift velocity (1.24 mm/μs for an electric field of 350 V/cm), it is straightforward to obtain the single point spatial resolution.

To get the valid set of data to be processed by the neural network algorithm or by the best-fit procedure, we use a quick mathematical filter which relies on the detection changes in the variance of the signal. A possible extension is to implement a neural network filter, as recently suggested in Ref. [8].

The results for different approaches are shown in Table 2. The single point spatial resolution measured using the hit finding procedure alone is improved by the neural network algorithm to a value not far from that obtained by the best-fit method. This is evident in Fig. 5, where the bidimensional image of the track of a cosmic muon crossing event is plotted using three different methods of analysis. The two coordinates are proportional to the TPC channel number and to the drift time respectively. The grey level of the pixel codes the pulse height, proportional to the detected charge. The increased ionization near a secondary interaction point along the track is easily noticed.

4.4. Computing time

The CPU time needed to process a complete TPC image (192 channels), using the above mentioned approaches, is shown in Table 3. From the table one can see that the neural network technique is one order of magnitude faster than the best-fit one. The computational time depends little on the architecture of the network and it is dominated by the input/output procedures and by the hit-finding algorithm. Considering the network simulation alone, we find that it is two orders of magnitude faster than the minimization process.

Table 3
CPU time values taken to process a complete TPC image

I/O procedures	0.94 s
I/O + hit finding procedure	1.81 s
I/O + hit finding procedure + neural network (200 h.u.)	2.15 s
I/O + hit finding procedure + neural network (100 h.u.)	2.01 s
I/O + hit finding procedure + neural network (50 h.u.)	1.95 s
I/O + hit finding procedure + best-fit	21.46 s

5. Conclusions

We find that the accuracy offered by an artificial neural network to extract physical information from TPC signals, such as those provided by the ICARUS 3 ton prototype, is good enough for this specific problem. The approach offers a good compromise between processing time and output resolution.

The implementation of a neural network algorithm on a sequential scalar computer offers three main advantages with respect to the standard best-fit procedure:
- the network approach is much faster, even when the simulation is performed with a sequential scalar computer;
- the process of extracting information can be limited to the few parameters of physical interest;
- the process of extracting information is not bounded to an analytical line shape and in principle it is possible to train the network using Monte Carlo data or even real signals.

Nevertheless this approach presents some weak points:
- it is not possible to build a statistical estimator whose value is intended as a meaningful guess for the unknown value of a parameter, or define a "confidence level" as in the normal best-fit procedures;
- the mathematical information stored in artificial neural networks is not directly controllable. It is very hard to locate possible error points and made corrections. The only way to do this is to perform a new training with a more precise example set.

The main disadvantage related to the use of a sequential scalar computer is that the training time is proportional to the total number of weights to be optimized; this limits the number of possible network configurations to be studied. Recently it has been proposed (see Ref. [9]) to implement a neural network algorithm of our type on a massive parallel computer (APE-100). This certainly would speed-up the calculations and make possible the use of more suitable, but more complex, network architecture.

The present neural network approach is particularly convenient when one needs a fast event analysing algorithm, as in the case of the on-line event selection foreseen for the ICARUS experiment [1]. The information from the network could be used to recognize the main features of the signals allowing a real-time event identification. The detector readout should be divided into a number of identical and independent subsystems with dedicated CPUs that accomplish the neural network simulation. In view of possible future application of this new technology in the ICARUS experiment further research and development work is needed.

References

[1] ICARUS Collaboration, ICARUS II, a second generation proton decay experiment and neutrino observatory at the Gran Sasso Laboratory, proposal Vol. I (22 September 1993) and Vol. II (21 May 1994).

[2] P. Benetti et al., Nucl. Instr. and Meth. A 332 (1993) 395.

[3] P. Benetti et al., Nucl. Instr. and Meth. A 345 (1994) 230.

[4] R. Hecht-Nielsen, in: Neurocomputing (Addison-Wesley, 1990).

[5] A. Galli Tognota, Degree Thesis (Pavia 1994).

[6] D.E. Rumelhart and J.L. McClelland, in: Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1 (MIT Bradford Press, Cambridge, 1986); M. McCord Nelson and W.T. Illingworth, in: A Practical Guide to Neural Nets, (Addison-Wesley, 1991).

[7] M. Goossens and F. James, MINUIT – Function minimization and Error Analysis (Version 92.1), Program Library D 506, CERN (1992).

[8] P. Hörnblad et al., Nucl. Instr. and Meth. A 336 (1993) 285.

[9] R. Tripiccione, Int. J. Mod. Phys. C 4 (1993) 425; P.S. Paolucci, INFN internal report ROM-NI-1017 (Rome 1993).